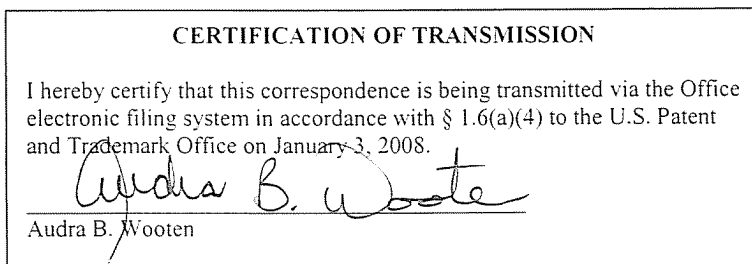


IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re:	Brabson et al.	Confirmation No.: 3407
Serial No.:	10/007,581	Group Art Unit: 2135
Filed:	December 5, 2001	Examiner: Joseph T. Pan
For:	OFFLOAD PROCESSING FOR SECURITY SESSION ESTABLISHMENT AND CONTROL	

Date: January 3, 2008

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450



APPELLANTS' REPLY BRIEF
ON APPEAL UNDER 37 C.F.R. §41.41

Sir:

This Reply Brief is filed in response to the Examiner's Answer mailed November 5, 2007 (hereinafter "Examiner's Answer"). It is not believed that an extension of time and/or additional fee(s) are required, beyond those that may otherwise be provided for in documents accompanying this paper. In the event, however, that an extension of time is necessary to allow consideration of this paper, such an extension is hereby petitioned for under 37 C.F.R. §1.136(a). Any additional fees believed to be due in connection with this paper may be charged to Deposit Account No. 50-0220.

I. The Examiner's Answer – Response to Argument

Appellants hereby incorporate herein the arguments set out in Appellants' Brief on Appeal filed July 25, 2007 (hereinafter "Appeal Brief") as if set forth in their entirety. In the interest of brevity, Appellants will only address new arguments made in the Examiner's Answer.

A. Section 10(i) of the Examiner's Answer responds to Appellants' showing at Section II.A. of the Appeal Brief that U.S. Patent No. 6,141,705 to Anand et al. ("Anand") is directed to a system by which security processing is performed by an offload component (e.g. a network interface card, or NIC) under the direction and control of an application program, not an operating system kernel, as recited in Claim 1. In particular, the Examiner's Answer cites col. 9, lines 11-15 of Anand as disclosing that the transport protocol driver, which Anand describes as controlling the NIC, is in the operating system kernel. The cited passage of Anand reads as follows:

In Windows NT, a transport protocol driver is a software component that implements a transport driver interface (TDI), or possibly another application-specific interface at its upper edge, to provide services to users of the network.

Anand, col. 9, lines 11-15. Appellants respectfully submit that this passage does not indicate that a transport protocol driver is part of a Windows NT kernel, but rather merely describes the logical location of a transport protocol driver within a Windows NT environment.

In the Appeal brief, Appellants have cited to numerous other passages of Anand that clearly distinguish between a transport protocol driver and the operating system kernel. See Anand, col. 8, lines 26-31 ("Each driver, typically implemented as a software component provided by the vendor of the corresponding NIC, is responsible for sending and receiving packets over its corresponding network connection and for managing the NIC on behalf of the operating system."). See, also, Anand col. 8, lines 50-55 ("More particularly, NDIS describes the interface by which one or multiple NIC drivers (116-120) communicate with one or multiple underlying NICs (100-104), one or multiple overlying transport protocol drivers, or transports, (represented at 128-134 in FIG. 2), and the operating system." (Emphasis added)

Thus, the Examiner's Answer has failed to rebut the Appellants' showing that operation of the offload component of Anand is not controlled by the operating system kernel.

B. Section 10(ii) of the Examiner's Answer responds to Appellants' showing at Section II.A. of the Appeal Brief that Figure 3 of Anand does not teach, and in fact excludes, kernel-based offload security processing as recited in Claim 1. Anand discloses a Windows NT architecture. As explained above, Anand clearly distinguishes between the Windows NT

operating system and the transport protocol driver, which controls NIC operation. In support of the Examiner's position, the Examiner's Answer cites Maurice Bach, "The Design of the UNIX Operating System," i.e., a reference apparently describing the UNIX operating system. A copy of Bach, which appears to be a newly cited reference, was not provided with the Examiner's Answer. Nevertheless, Appellants submit that a UNIX reference is not relevant to the question of whether Anand discloses that a transport protocol driver is part of a Windows NT operating system kernel. Accordingly, the Examiner's Answer has not rebutted the Appellants' showing that Figure 3 of Anand does not teach kernel-based offload security processing.

C. Sections 10(iii) and 10(iv) of the Examiner's Answer respond to Appellants' showing at Section II.A. of the Appeal Brief that the passage of Anand at col. 3, lines 9-23, does not mean that in the system of Anand, an operating system kernel can initiate a request to a security offload component to secure a particular communication with a remote unit in response to receiving a request from an application program to initiate a communication with the remote unit, as recited in Claim 1.

The Examiner's Answer appears to concede that Anand's teaching is deficient in this regard, but cites U.S. Publication No. 2003/0014623 to Freed et al. ("Freed"). Freed teaches a method of managing a communication negotiation between a client and a server. In particular, the Examiner's Answer cites paragraph [0068] of Freed, which states as follows (emphasis added):

In a manner similar to the embodiment shown in FIG. 5, the client 100 sends a handshaking packet SYN packet to TCP port 443 of SSL accelerator 250 rather than directly to server 300 (at step 202a) The SSL Accelerator device 250 will receive (at step 204a) the SYN packet transmitted by client 100 and may perform functions on packet to enable the SSL acceleration device to continue to perform its SSL proxy functions.

The Examiner's Answer concludes that "a combination of Anand and Freed disclose an operating system kernel can initiate a request to a security offload component to secure a particular communication with a remote unit in response to receiving a request from an application program to initiate a communication with the remote unit, as recited in Claim 1." Examiner's Answer, page 18.

Appellants respectfully submit that the cited passage of Freed simply does not teach what the Examiner's Answer contends. Instead, it refers to a **client** (i.e. **not** an operating system kernel) sending a handshaking packet to an SSL accelerator. The accelerator receives the packet transmitted by the client and performs functions on the packet. Nothing in the cited passage indicates that an operating system kernel can initiate a request to a security offload component to secure a particular communication with a remote unit in response to receiving a request from an application program to initiate a communication with the remote unit. Thus, even if combined with Anand, the combination would not teach or suggest that an operating system kernel can initiate a request to a security offload component to secure a particular communication with a remote unit in response to receiving a request from an application program to initiate a communication with the remote unit, as recited in Claim 1. Accordingly, the Examiner's Answer has failed to rebut the Appellants' showing in this regard.

D. Section 10(v) of the Examiner's Answer responds to Appellants' showing at Section II.A. of the Appeal Brief that the Examiner's interpretation of the passage of Anand at col. 13, lines 44-56 clearly ignores the term "environment" in the phrase "Windows NT environment." In response, the Examiner's Answer merely cites to column 3, lines 45-50 of Anand, which states:

In a preferred embodiment of the present invention, in the Windows NT layered networking architecture, a transport protocol driver, or transport, is implemented with an appropriate program method so as to be capable of querying each of the device driver(s) associated with the corresponding NIC(s) connected to the computer.

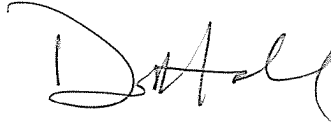
However, in this passage, Anand is describing the Windows NT **networking architecture**, not a Windows NT operating system kernel. In the context of software and as used in the cited passage of Anand, an "architecture" refers to the manner in which the components of a computer or computer system are organized and integrated. It is not used to describe an operating system kernel. Moreover, as explained above, Anand repeatedly and consistently distinguishes between the operating system on one hand and the transport protocol driver on the other hand.

E. Section 10(vi) of the Examiner's Answer responds to Appellants' showing at Section II.A. of the Appeal Brief that Anand does not suggest the benefits of controlling the operation of a security offload component using a control function in an operating system kernel rather than requiring such control functionality to be implemented in an application program. The Examiner's Answer simply refers to its earlier conclusions, and does not point to any specific teaching of Anand that suggests the benefits of controlling the operation of a security offload component using a control function in an operating system kernel. Accordingly, the Examiner's Answer has failed to rebut the Appellants' showing in this regard.

II. Conclusion

For at least the reasons set forth above and in Appellants' Appeal Brief, Appellants request reversal of the rejections of the pending claims, allowance of the pending claims, and passing of the application to issue.

Respectfully submitted,

A handwritten signature in black ink, appearing to read 'D. Hall', with a stylized flourish at the end.

David C. Hall
Registration No. 38,904

Myers Bigel Sibley & Sajovec, P.A.
P. O. Box 37428
Raleigh, North Carolina 27627
Telephone: (919) 854-1400
Facsimile: (919) 854-1401
Customer No. 46589